

An HTML 5 Primer

prepared by

Michael Greenberg

HTML 5 and HTML 4^[1]

What's the difference?

- Simple DOCTYPE declaration
- Elements which improve document structure
 - Ex: <section>, <article>, <aside>, <header>, <footer>, <nav>, <dialog>, <figure>, <details>, <progress>
- Elements which facilitate or add functionality
 - Ex: <audio>, <video>, <canvas>, <datagrid>, <datalist>
- New <input> types
 - Ex: datetime, date, month, week, time, number, range, email, url
- ping attribute for <a> and <area> elements
- charset attribute for <meta> element

HTML 5 and HTML 4

What's the difference? (continued...)

- form attribute for `<input>`, `<output>`, `<select>`, `<textarea>`, `<button>`, `<fieldset>` elements
- `manifest` attribute for `<html>` element supporting client-side offline applications
- New APIs to aid Web App development
 - Worker, 2-D Drawing, History, Offline Cache, and then some...
- Removed presentation-only attributes
 - Ex: `align`, `bgcolor`, `cellpadding/cellspacing`, `frame/frameborder`, `nowrap`, `valign`, `width/height`

Making an HTML5-compliant Doc

Simple DOCTYPE means creating HTML documents which degrade gracefully into a “standards-compliant mode” in browsers REGARDLESS of which version of HTML is being used.^{[2][3]}

HTML 5

```
<!DOCTYPE html>  
...  
</html>
```

..or..

XHTML 1.1

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML  
1.1//EN"  
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">  
...  
</html>
```

Straight-forward charset attribute makes it easier to have your documents properly displayed in the needed character set.

HTML 5

```
<meta charset="UTF-8">
```

..or..

not HTML 5

```
<meta http-equiv="Content-Type"  
content="text/html; charset=UTF-8">
```

Semantic Elements in HTML 5

`<section>`, `<article>`, `<aside>`, `<header>`, `<hgroup>`,
`<footer>`, `<nav>`, `<dialog>`, `<figure>`, `<time>`, `<mark>`

- New elements which separate sections of content on the page and provide additional meaning to the browser
- Removes need for ID attributes necessary to identify specific portions of markup
- Correctly use these elements which previously required a “hacked” use of markup tags
 - `<div id=“header”>` or ``
- Adds value to your HTML markup by giving apps the ability of parsing the important content apart from everything else.

Semantic Elements in HTML 5

Caveats to note in Internet Explorer (surprised?)

- Internet Explorer (in any version) does not correctly insert these elements into the DOM^[4]

```
<article>
  <h1>Title</h1>
  <p>Article Body with <span>span text</span>.</p>
</article>
```

HTML Example Page

```
<article>
|
|
<h1>
|
+--textNode("Title")
<p>
|
+--textNode("Article Body with ")
+--<span>
|   +--textNode("span text")
|
+--textNode(".")
```

Internet Explorer's DOM

```
<article>
|
+--<h1>
|   +--textNode("Title")
|
+--<p>
|   +--textNode("Article Body")
+--<span>
|   +--textNode("span text")
|
+--textNode(".")
```

Correct DOM

Semantic Elements in HTML 5

Caveats to note in Internet Explorer (surprised?)

- Get Internet Explorer to easily recognize these (and other) elements in your HTML with the simple fix illustrated below.

```
<!--[if IE]> Internet Explorer Fix for HTML5
<script>
  var e = "abbr,article,aside,audio,canvas,datalist,details," +
          "figure,footer,header,hgroup,mark,menu,meter,nav,output," +
          "progress,section,time,video".split(', ');
  var i = e.length;
  while (i--)
    { document.createElement(e[i]); }
</script>
<![endif]-->
```

- Remy Sharp provides this code in a convenient minified format for inclusion in your HTML.

<http://remysharp.com/2009/01/07/html5-enabling-script/>

The `<datalist>` and `<datagrid>` Element

- Assists form elements and data displays
- `<datalist>` provides an enumerated list of values. This can be connected to a freeform text input using the “**list**” attribute.
- `<datagrid>` generates a tree-styled set of data which can be interactive via an API

The <audio> and <video> Elements

- Native support for inline audio and video now supported in HTML 5.
- Native API to get status of multimedia and hooks for controlling playback.
- Browser compatibility between formats are still an issue.

| | Ogg Vorbis | MP3 | WAV |
|--------------|-------------|-----|-----|
| Firefox 3.5+ | ✓ | - | ✓ |
| Safari 4+ | - | ✓ | ✓ |
| Chrome 3+ | ✓ | ✓ | - |
| Opera 10+ | - | - | ✓ |
| IE | No support. | | |

Audio Format Support

| | Theora Vorbis | H.264 |
|--------------|---------------|-------|
| Firefox 3.5+ | ✓ | - |
| Safari 3+ | - | ✓ |
| Chrome 3+ | ✓ | ✓ |
| Opera 10+ | - | ✓ |
| IE | No support. | |

Video Format Support

The <audio> Element

Attributes

- **src** contains a valid URL to the audio content
- **controls** tells the browser to display its default controls
- **autobuffer** tells the browser to begin downloading the media immediately when the page is loaded
- **autoplay** tells the browser to start playing the media as soon as it can
- **loop** tells the browser if it should start again once the media playback has ended

Methods

- **play()** starts media playback
- **pause()** pauses media playback
- **canPlayType(*mime-type*)** asks the browser to verify if it supports the given mime-type

* As of version 10, **Opera** only supports the **Audio()** object and **play()** method.

Simple Example

```
<audio src="theRaven_Poe.ogg" controls autobuffer>
  <a href="theRaven_Poe.ogg" title="Download the Sample">OGG</a>
  <a href="theRaven_Poe.mp3" title="Download the Sample">MP3</a>
  <a href="theRaven_Poe.wav" title="Download the Sample">WAV</a>
</audio>
```

The <audio> Element

- <source> allows multiple media files to be used to improve compatibility
 - The only change is that the **src** attribute is now placed within the <source> element.
 - A flash-based solution never hurts either.
- **Firefox** needs its file FIRST!
(Firefox cannot degrade to the next <source> element if the first one fails.)
- Could this possibly be easier?
<http://www.happyworm.com/jquery/jplayer/>

```
<audio controls autobuffer>
  <source src="theRaven_Poe.ogg" />
  <source src="theRaven_Poe.mp3" />
  <source src="theRaven_Poe.wav" />
  <!-- Flash solution here for complete support. -->
  <!-- If no flash, then at least the links will show. -->
  <a href="theRaven_Poe.ogg" title="Download the Sample">OGG</a>
  <a href="theRaven_Poe.mp3" title="Download the Sample">MP3</a>
  <a href="theRaven_Poe.wav" title="Download the Sample">WAV</a>
</audio>
```

Improved Example

The <video> Element

Attributes

- The same attributes as <audio>, plus...
- **height** and **width** sets the size of the video window

Methods

- The same methods as <audio>
- REALLY annoying type attribute^[6]. Must describe container AND codec within the attribute value

Type Attribute Example

```
<source src="pr6.mp4" type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"' >
```

- Still, saves on bandwidth when browser ignores unsupported files

Caveats

- Unlike with the <audio> element, **Firefox** doesn't care which source comes first!
- IE needs some special help. (Surprised?)
<http://diveintohtml5.org/video.html#ie>

The <canvas> Element

- Arguably the single most complicated element officially supported!
- Not new, but since adoption in HTML5 we can expect much better support and adoption.
- Experimental 3D support in Opera^[7] and Firefox^[8] with varying implementations

**** Internet Explorer** needs an external script (**excanvas.js**) to support <canvas> elements^[9]

**** Internet Explorer** is SLOW with this configuration.

| | Canvas Support |
|------------|-------------------------|
| Firefox 3+ | ✓ |
| Safari 3+ | ✓ |
| Chrome 2+ | ✓ |
| Opera 9.2+ | ✓ |
| IE | External Support Needed |

The `<canvas>` Element

Testing Compatibility

- Using `getContext(' 2d')` we can test the browsers ability to support `<canvas>` methods.
- Calling this method prepares the browser to draw on the area.

```
$(function () {  
    var obj = $("#ourCanvasEx")[0];  
    if (obj.getContext) {  
        var context = obj.getContext("2d");  
        if (context) {  
            context.strokeRect(25,25,250,100);  
            context.textAlign = "center";  
            context.fillText("It works!",150, 75);  
        }  
    }  
});
```

Compatibility Test Example

The `<canvas>` Element

Drawing Lines and Shapes

Attributes (all within the '2d' context)

- **fillStyle** accepts #hex, rgb(), and rgba() and gradients
- **strokeStyle** accepts the same as above
- **lineWidth** accepts an integer as width in pixels
- There are more, but this is a good start!

Methods (all within the '2d' context)

- **fillRect(x, y, width, height)** will make a filled in rectangle
- **strokeRect(x, y, width, height)** will make an outlined rectangle
- **clearRect(x, y, width, height)** will erase all contents within the dimensions given
- **moveTo(x, y)** moves your virtual pen to the canvas's coordinates
- **lineTo(x, y)** draws a line from the previous location to the new location
- **fill()** fills in within the virtual lines drawn using the above functions
- **stroke()** draws just the lines using the above functions
- **closePath()** will draw a virtual line back to the first point

The `<canvas>` Element

Drawing Text

Attributes (all within the '2d' context)

- **font** accepts anything that would go in the CSS "font" rule (font, variant, weight, size...)
- **textAlign** accepts **start**, **end**, **left**, **right** and **center** which all do what you'd expect (left = start, right = end)
- **textBaseline** accepts **top**, **hanging**, **middle**, **alphabetic**, **ideographic**, or **bottom**

Methods (all within the '2d' context)

- **fillText(text, x, y)** will draw text at the coordinates (x,y)

The <canvas> Element

Drawing Gradients

Methods (all within the '2d' context)

- **createLinearGradient(x0, y0, x1, y1)** paints along a line from (x0, y0) to (x1, y1).
- **createRadialGradient(x0, y0, r0, x1, y1, r1)** paints along a cone between two circles. The first three parameters represent the start circle, with origin (x0, y0) and radius r0. The last three parameters represent the end circle, with origin (x1, y1) and radius r1.
- **addColorStop(0<n<1, color)** adds a transitional color in the gradient. (This method acts on the object created by the above two methods.)

**** Internet Explorer with `excanvas.js` does not support Radial Gradients!**

```
// Within a context
var grad = context.createLinearGradient(0,0,300,0);
grad.addColorStop(0,"black");
grad.addColorStop(0.5,"red");
grad.addColorStop(1,"blue");
context.fillStyle = grad;
context.fillRect(0,0,100,100);
```

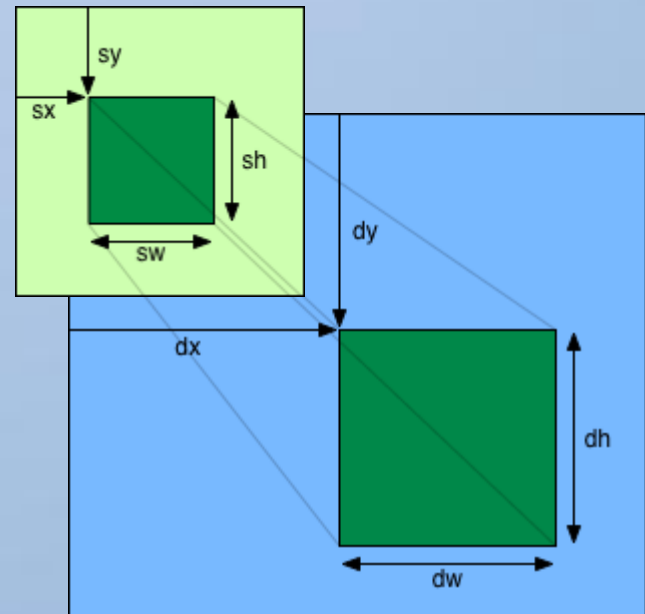
[Gradient Example](#)

The `<canvas>` Element

Drawing Images

- Drawing images uses a complex method which can be used in three different ways
 - `drawImage(img_elem, dx, dy)`
 - `drawImage(img_elem, dx, dy, dw, dh)`
 - `drawImage(img_elem, sx, sy, sw, sh, dx, dy, dw, dh)`
- `Img_elem` is the `` or `<canvas>` element to draw.

** Internet Explorer with `excanvas.js` does not correctly scale strokes from other canvases!



The `<canvas>` Element

Drawing Images

- Can also manipulate images pixel-by-pixel:
 - `createImageData(w, h)` creates an object to draw a picture
 - `getImageData(sx, sy, w, h)` can get the pixel data from a canvas object
 - `putImageData(img_data, dx, dy)` draws the pixel data onto the canvas in the coordinates specified
- These methods use an `ImageData` object which is defined

```
{  width:w,  height:h,  data: array(r,g,b,a,...) //size = (w*h*4)}

```

The `<canvas>` Element

Great Examples and Demonstrations

- **Basic Image Inversion Filter**
<http://www.robodesign.ro/coding/canvas-primer/20081208/example-imagedata.html>
- **Mandelbrot Fractal Generator**
http://jwf.us/blog_apps/mandelbrot_canvas/
- **Torus**
<http://www.benjoffe.com/code/games/torus/>
- **Twitter Visualization**
<http://9elements.com/io/?p=153>
- **MORE!**
<http://www.canvasdemos.com/>

The **manifest** Attribute

Offline Caching

- The page along with files within the **MANIFEST** will be downloaded and will not be re-cached until the **MANIFEST** is changed!
- The **MANIFEST** attributes points to a relative file which identifies which files should be included.
- Directives to be used within the **MANIFEST**
 - **CACHE**: is used to specify a file's relative path for local caching. (default)
 - **NETWORK**: specifies that these files should ALWAYS be sources from the network
 - **FALLBACK**: specifies files to substitute should a file within the given namespace is unavailable
- **Example:** <http://html5demos.com/offlineapp>
- ** **NOTE:** For this demo to work, it must be retrieved from the network. Needs Safari 4+ or Firefox 3.5.3+

Database API

- Very young implementation and currently only works in specialized builds of webkit-based browsers (Safari 4+, iPhone and iTouch OS 2.2)
- Most implementations in a SQLite storage engine.

Methods

- **openDatabase(name, version, display_name, size)** initializes a database in the browser's local cache.
- **executeSql(sql, data, callback)** executes a SQL statement within the opened database
** **NOTE:** Synchronous call
- **DB Storage Tutorial**
<http://www.weboshelp.net/webos-tutorials/156-palm-webos-html5-database-storage-tutorial>
- **Third-party Demo:**
http://code.google.com/apis/gears/samples/hello_world_database.html

Thank you!

- Plenty of resources available as the technology matures. (See References in this Presentation)
- This presentation is available for download with examples.
<http://nobulb.com/2009/11/html-5-primer/>
- Corrections and suggestions via email
mg@nobulb.com

References

1. **HTML 5 differences from HTML 4**
<http://www.w3.org/TR/2008/WD-html5-diff-20080122/>
2. **HTML5 DOCTYPE**
<http://ejohn.org/blog/html5-doctype/>
3. **An explanation of DOCTYPEs**
<http://diveintohtml5.org/semantics.html#the-doctype>
4. **A Brief Discussion on How Browsers Handle Unknown Elements**
<http://diveintohtml5.org/semantics.html#unknown-elements>
5. **Native Audio in the browser**
<http://html5doctor.com/native-audio-in-the-browser/>
6. **<video> "type" parameters**
http://wiki.whatwg.org/wiki/Video_type_parameters
7. **Opera 3D support for <canvas>**
<http://my.opera.com/timjoh/blog/2007/11/13/taking-the-canvas-to-another-dimension>

(More) References

8. **Firefox 3D support for <canvas>**
<http://blog.vlad1.com/2007/11/26/canvas-3d-gl-power-web-style/>
9. **Internet Explorer <canvas> support (ExCanvas)**
<http://code.google.com/p/explorercanvas/>
10. **WHATWG specification on <canvas>**
<http://www.whatwg.org/specs/web-apps/current-work/multipage/the-canvas-element.html#the-canvas-state>
11. **Collection of HTML 5 Demos**
<http://html5demos.com/>